Synthetic topology in Homotopy Type Theory for probabilistic programming

Florian Faissole[1] Bas Spitters[2]

[1]Inria - LRI

[2]Aarhus

Discrete probabilities : ALEA library

ALEA library (Audebaud, Paulin-Mohring) basis for CertiCrypt

- Discrete measure theory in Coq;
- Monadic approach (Giry, Jones/Plotkin, ...):
  - CPS: $\overbrace{\underbrace{(A \to [0,1])}_{\text{'}meas.\ functions\text{'}} \quad \to [0,1]}^{\text{'}measures\text{'}}$
  - submonad: monotonicity, summability, linearity.
    Coq cannot prove that this is a monad (no funext).

Example: flip coin : $M bool$

$$\lambda\ (f : bool \to [0,1]).(0.5 \times f(true) + 0.5 \times f(false))$$

First question: Can we avoid 'setoid hell'?

Discrete probabilities : ALEA library

ALEA library (Audebaud, Paulin-Mohring) basis for CertiCrypt

- Discrete measure theory in Coq;
- Monadic approach (Giry, Jones/Plotkin, ...):
  - CPS: $\overbrace{\underbrace{(A \to [0,1])}_{\text{'}meas.\ functions\text{'}} \to [0,1]}^{\text{'}measures\text{'}}$
  - submonad: monotonicity, summability, linearity.
    Coq cannot prove that this is a monad (no funext).

Example: flip coin : $M bool$

$$\lambda\ (f : bool \to [0,1]).(0.5 \times f(true) + 0.5 \times f(false))$$

First question: Can we avoid 'setoid hell'?
Extend with $[0,1]$? For machine learning (augurV2)
For differential privacy: aprhl version of easycrypt

Univalent homotopy type theory

Coq lacks quotient types and functional extensionality.
ALEA uses setoids, $(T, \equiv)$.

Univalent homotopy type theory: an internal type theory for a generalization of setoids. We use Coq's HoTT library.
(CPP: Bauer, Gross, Lumsdaine, Shulman, Sozeau, Spitters)

Desiderata for a framework for the semantics of probabilistic programming languages?

Such a framework should support:

- continuous and discrete types, and a rich variety of functions between them,
- random choice $+_r$,
- choosing from a rich variety of discrete and continuous distributions,
- conditioning,
- a rich type system including sum, product, function, and probability types,
- recursive definitions of values, and
- recursive definitions of types.

Probability theory

How to formalize probability?

- Classical probability: measures on σ-algebras *of sets*
  σ-algebra: collection closed under countable $\bigcup, \bigcap$
  measure: σ-additive map to $\mathbb{R}$.
- Giry monad in semantics:
  $X \mapsto Meas(X)$ is a monad
  on subcategories of topological spaces or domains.
  valuations restrict measures to opens.

Topologies in Coq ... ?

Synthetic topology

Scott: Synthetic domain theory
Domains as sets in a topos (Hyland, Rosolini, ...)
By adding axioms to the topos we make a DSL for domains.
Synthetic topology
(Brouwer, ..., Escardo, Taylor, Vickers, Bauer, ...)
Every object carries a topology, all maps are continuous
Idea: Sierpinski space $\Sigma = \left(\begin{smallmatrix}\odot \\ \cdot\end{smallmatrix}\right)$ classifies opens:

$$O(X) \cong X \to \Sigma$$

Convenient category of/type theory for 'topological' spaces.
Synthetic (real) computability
semi-decidable truth values Σ classify semi-decidable subsets.

Common generalization based on abstract properties for Σ:
Dominance axiom: maps classified by Σ compose.

More axioms for synthetic topology

Let $N^\bullet$ be the type of increasing binary sequences
'the one-point compactification of N'.
WSO ('Weakly Sequentially Open'):
The intrinsic topology, $N^\bullet \to \Sigma$, coincides with the metric topology $d(\underline{n}, \underline{m}) = 2^{-\min(n,m)}$.
If $f : N^\bullet \to \Sigma$ and $f(\infty) = 1$, then there exists n s.t. $f(n) = 1$.
WSO contradicts classical logic, but holds in our models.
A stronger principle:
Fan: $2^{\mathbb{N}}$ is metrizable and compact

Lešnik developed much analysis synthetically from these principles.
Countable choice is often not needed.

Models for synthetic topology

Internal language of a topos: IHOL/ dependent predicate logic.
Standard axioms for continuous computations:
Brouwer, Kleene realizability K_2 (TTE)
(Recently implemented in NuPrl)
Gives realizability topos

Synthetic topology:
Big toposes: sheaf toposes over a subcategory of Top, e.g. 0-ωTop
Embedding of realizability models into sheaf models
(Awodey/Bauer)

Toposes and types

How to formalize toposes in type theory?
Use HoTT as a language for higher toposes.
Rijke/S: hSets in HoTT form a (predicative) topos:
large power objects.

Conjecture (Shulman,...):
Both Grothendieck toposes and realizability can be lifted to HoTT
Partial results:
Simplicial sheaves (Cisinski/Shulman)
Cubical stacks (Coquand)
Cubical assemblies (CMU)
Cubical model in NuPrl (Bickford,Coquand,Mörtberg)

Here: we show how this is useful.
Our second use of HoTT.
Predicative constructive maths without countable choice.

Implementation in HoTT

Our basis: Cauchy reals in HoTT as HIIT (book, Gilbert)

- HoTTClasses: like MathClasses but for HoTT
  Math-classes: abstract approach to continuous computation in Coq, using type classes (Krebbers, Spitters, van der Weegen).
- Experimental Induction-Recursion branch by Sozeau

Partiality (Altenkirch, Danielson): Construction in HoTT:
free ω-cpo completion as a higher inductive inductive type:

$$A_\perp : hSet \qquad \perp : A_\perp \qquad \eta : A \to A_\perp$$

$$\subseteq_{A_\perp} : A_\perp \to A_\perp \to Type$$

$$\bigcup : \prod_{f:\mathbb{N}\to A_\perp} (\prod_{n:\mathbb{N}} f(n) \subseteq_{A_\perp} f(n+1)) \to A_\perp$$

$\subseteq$ must satisfy the expected relations.

$\mathbb{S}$:=Partial(1) as Σ.
Fits with ssreflect style reflection into semi-decidable propositions.

Implementation in HoTT

Prop: $\mathbb{S}$ is a dominance.
Escardó and Knapp provide a general construction of a lifting $\mathcal{L}$ defined by a dominance.
They ask: $X_\perp \equiv \mathcal{L}_\mathbb{S} X$?
We show that it is.
In fact, $X_\perp$ is a classifier for partial maps with open domain.
This uses univalence in the form of the object classifier.

Implementation in HoTT

Lower Reals (small):
$r : \mathbb{R}_l := \mathbb{Q} \to \mathbb{S}$
$\forall p, r(p) \iff \exists q, (p < q) \wedge r(q).$
$\rightsquigarrow$ lower semi-continuous topology.

Dedekind Reals (Gilbert):
$$\mathbb{R}_D := \underbrace{(\mathbb{Q} \to \mathbb{S})}_{lower\ real} \times \underbrace{(\mathbb{Q} \to \mathbb{S})}_{upper\ real}$$

Valuations:
Valuations on $A : hSet$:
$Val(A) = (A \to \mathbb{S}) \to \mathbb{R}_l^+$

- $\mu(\emptyset) = 0$
- Modularity
- Monotonicity
- Continuity

Integrals:
Positive integrals:
$Int^+(A) = (A \to \mathbb{R}_D^+) \to \mathbb{R}_D^+$

- $\int(\lambda x.0) = 0$
- Additivity
- Monotonicity
- Probability: $\int \lambda\_.1 = 1$

Riesz theorem: homeomorphism between integrals and valuations.
Constructive proof (Coquand/S): A regular compact.

Implementation in HoTT (2)

Lower Reals:
$r : \mathbb{R}_l := \mathbb{Q} \to \mathbb{S}$
$\forall p, r(p) \iff \exists q, (p < q) \wedge r(q).$
$\rightsquigarrow$ lower semi-continuous topology.

Dedekind Reals (Gilbert):
$$\mathbb{R}_D := \underbrace{(\mathbb{Q} \to \mathbb{S})}_{lower\ real} \times \underbrace{(\mathbb{Q} \to \mathbb{S})}_{upper\ real}$$

Valuations:
Valuations on $A : hSet$:
$Val(A) = (A \to \mathbb{S}) \to \mathbb{R}_l^+$

- $\mu(\emptyset) = 0$
- Modularity
- Monotonicity
- Continuity

Lower integrals:
Positive integrals:
$Int^+(A) = (A \to \mathbb{R}_l^+) \to \mathbb{R}_l^+$

- $\int(\lambda x.0) = 0$
- Additivity
- Probability
- Monotonicity

Riesz theorem: homeomorphism between integrals and valuations.
Constructive proof by Vickers: A locale. Here: synthetically.

Monadic semantics

Giry monad: (space) $\rightsquigarrow$ (space of its valuations):

- functor $\mathcal{M} : Space \to Space$.
- unit operator $\eta_x = \delta_x$ (Dirac)
- bind operator $(I >\!\!>\!= M)(f) = \underline{\int_I} \lambda x.(Mx)f$.

$(>\!\!>\!=) :: \mathcal{M}A \to (A \to \mathcal{M}B) \to \mathcal{M}B$.

Probabilistic languages

Functional languages: Rml (ALEA):
Standard monadic interpretation:

- $v \rightsquigarrow \mathsf{unit}(v)$
- let $x = a$ in $b \rightsquigarrow$ bind $[a]$ $(\lambda\ x.[b])$
- $f(e_1,\ldots,e_n) \rightsquigarrow$ bind $[e_1]$ $(\lambda\ x_1 \ldots$ bind $[e_n]$ $(\lambda\ x_n.[f](x_1,\ldots,x_n))$
- if b then c_1 else $c_2 \rightsquigarrow$ bind $[b]$ $(\lambda\ x{:}\mathbf{2}.$ if x then $[c_1]$ else $[c_2])$

Imperative language:

- pWhile (base language for Certi/Easy-crypt)
  $\rightsquigarrow$ usual While language + random assignment.

Function types

To interpret the full computational λ-calculus we need T-exponents $(A \to TB)$.
The standard Giry monads do not support this.
hSet is cartesian closed, so we obtain a higher order language.
Moreover, the Kleisli category is ω-cpo enriched (we use subprobability valuations), so we can interpret fixed points.

Rich programming language, as requested by Plotkin.

AugurV2

Huang (Morrisett)
Compiled probabilistic programming language geared towards machine learning
Higher order with recursion and continuous data types
Computable semantics based on topological domains
A related sampling semantics
Huang and I are proving that the semantics agree
using the abstract realizability approach to TTE (Bauer/Lietz/...)

HoTT as a verification framework?

Higher order probabilistic computation

Compare: Top is not Cartesian closed.
1. Define a convenient super category. E.g. quasi-topological spaces: concrete sheaves over compact Hausdorff spaces.
This is a quasi-topos which models synthetic topology.
Even: big topos
2. Add probabilities inside this setting.

Staton, Yang, Heunen, Kammar, Wood model for higher order probabilistic programming has the same ingredients (but in opposite direction):
1. Standard Giry model for probabilistic computation
2. Obtain higher order by (a tailored) Yoneda

Computation

We used axioms for HoTT and synthetic topology.
This breaks computation.
However, implemented in cubical and NuPrl, respectively.
Hope for an implementation supporting both, either cubical in NuPrl (Bickford), RedPrl, or along the lines of guarded cubical.

Computation

We used axioms for HoTT and synthetic topology.
This breaks computation.
However, implemented in cubical and NuPrl, respectively.
Hope for an implementation supporting both, either cubical in NuPrl (Bickford), RedPrl, or along the lines of guarded cubical.

General fixed points, so no guarantee on termination (as in ALEA)
But, for a program p with randomness from $[0, 1]$ the semantics:

$[\![p]\!](I) > r$, where I is a rational interval in $[0, 1]$ and $r \in \mathbb{Q}$.

will be semi-decidable.

Non-measurable sets

How can this work???

Classically, $\Sigma = 2$, so all sets need to be measurable.
AC then implies no Lebesgue measure on [0,1].

Continuously, there are fewer maps $[0, 1] \to \Sigma$, hence we can model the Lebesgue valuation.

Conclusions

- Probabilistic computation with continuous data types
  axiomatic semantics a la ALEA
- HoTT replaces setoids
- Experiment with synthetic topology in HoTT
- Extension of the Giry monad from locales to synthetic topology
- Model for higher order probabilistic computation
  AugurV2
- Formalization in Coq (WIP)
  `https://github.com/FFaissole/Valuations`